Status of the Claims.

1. (canceled).

2. (previously presented) The method according to claim 27, further including receiving as input event types, business component types, and dependency types associated with a business domain.

3. (previously presented) The method according to claim 27, further including receiving as input rules that describe how a given event affects a specified business component.

4. (previously presented) The method according to claim 27, further including receiving as input rules that describe when a change in a business component triggers an event.

5-7. (Canceled)

8. (previously presented) The method according to claim 27, wherein the definition includes predefined dependency type semantics.

9. (previously presented) The method according to claim 8, wherein said dependency type semantics include a mandatory logical operator that logically couples one or more source components of the dependency to one or more targets of the dependency and sets the targets to a worst state of the source components.

10. (previously presented) The method according to claim 8, wherein said dependency type semantics include an "N out of M" logical operator, wherein N is less than M, that logically couples M source components of the dependency to one or more targets of the dependency and sets the targets to ok if at least N of the source components are ok and otherwise sets the targets to "fail".

11-26. (Canceled).

27. (currently amended) A method for processing information, comprising:

in a system comprising one or more processors, providing an active dependency integration unit, comprising a first program module that receives as input first events for processing together with a definition of dependencies between business components in a business model in order to monitor a propagated impact between the business components;

providing in the system a situation awareness unit ~~management unit~~, comprising a second program module that detects situations comprising specified combinations of second events and conditions;

receiving in the active dependency integration unit a first event relating to at least a first business component;

responsively to the first event and to the dependencies, propagating a change to at least a second business component;

passing a second event indicative of the change to the situation awareness unit;

responsively to the second event, detecting a situation in the situation awareness unit;

responsively to the situation, conveying a third event from the situation awareness unit to the active dependency integration unit; and

outputting a functional state of the business model responsively to at least the third event.

28. (new) A computer software product, including a computer-readable storage medium in which computer program instructions are stored, wherein the instructions comprise distinct software modules including an active dependency integration unit and a situation awareness unit, and when executed by a processor, the instructions cause the processor to perform a method for processing information, comprising:

detecting in the situation awareness unit situations comprising specified combinations of events and conditions relating to a business model, the conditions comprising an order of occurrence and temporal relationships among the events;

receiving as input in the active dependency integration unit events relating to business components in the business model, for processing together with a definition of dependencies among the business components in order to monitor a propagated impact of the events among the business components, including receiving a first event relating to a first business component;

responsively to the first event and to the dependencies, propagating a second event indicative of a change to at least a second business component, wherein the dependencies between the first business component and the second business component comprise a compound dependency having two different simple dependency types;

passing the second event to the situation awareness unit;

responsively to the second event, detecting a situation in the situation awareness unit;

responsively to the situation, conveying a third event from the situation awareness unit to the active dependency integration unit; and

outputting a functional state of the business model responsively to at least the third event.

29. (new) The computer software product according to claim 28, wherein the simple dependency types are a mandatory dependency and a disjunctive dependency.

30. (new) A method for processing information, comprising:

executing in a processor distinct program modules including an active dependency integration unit and a situation awareness unit to cause the processor to perform the steps of:

detecting in the situation awareness unit situations comprising specified combinations of events and conditions relating to a business model, the conditions comprising an order of occurrence and temporal relationships among the events;

receiving as input in the active dependency integration unit events relating to business components in the business model, for processing together with a definition of dependencies among the business components in order to monitor a propagated impact of the events among the business components, including receiving a first event relating to a first business component;

responsively to the first event and to the dependencies, propagating a second event indicative of a change to at least a second business component, wherein the dependencies between the

first business component and the second business component comprise a compound dependency having two different simple dependency types;

passing the second event to the situation awareness unit;

responsively to the second event, detecting a situation in the situation awareness unit;

responsively to the situation, conveying a third event from the situation awareness unit to the active dependency integration unit; and

outputting a functional state of the business model responsively to at least the third event.

31. (new) The method according to claim 30, wherein the simple dependency types are a mandatory dependency and a disjunctive dependency.